Exhibit 2

Matching 2D Patterns of Protein Spots *

Frank Hoffmann† Klaus Kriegel† Carola Wenk†

Department of Computer Science
Freie Universität Berlin

Abstract

A new algorithmic approach to comparing 2D patterns of protein spots obtained by the 2D gel electrophoresis technique is presented. Both the matching of a local pattern vs. a full 2D gel image and the global matching between full images are discussed. Preset slope and length tolerances of pattern edges serve as matching criteria. The local matching algorithm relies on a data structure derived from the incremental Delaunay triangulation of a point set and a 2–step hashing technique. The approach for the global matching uses local matching for landmark settings, which in most previous algorithmic solutions has been done interactively by the user.

1 Introduction

1.1 Point Pattern Matching

The matching problem for geometric point patterns has been subject of intensive research in the last decade. Given a point pattern P and another target point set T one wants to compute all occurrences of P in T. Usually, an admissible space $\mathcal{A}$ of transformations (e.g. translations, rigid motions and/or scalings) is given which can be used to map the pattern into or as close as possible to the point set T. Most common is to consider the Hausdorff H distance, see [10]. Additionally we have a distance measure d between patterns. In general, we want to find such an $f \in \mathcal{A}$ and a pattern Q in T for which $d(f(P), Q) \leq \epsilon$, where ϵ is a prescribed error tolerance. We distinguish between exact matchings (ϵ is zero) and approximate matching solutions, otherwise. The latter are important in most practical applications. Like in our concrete application it is sometimes only possible to find partial matchings, i.e., we will be looking for as large as possible sub-patterns of P which have an approximate matching pattern in T.

A survey on several variants of the general geometric matching problem, different geometric approaches, and various algorithmic techniques can be found in [2]. In particular, for most settings there are algorithms which solve the problem well from a theoretical point of view but which are hardly applicable in practice because of their time complexity.

Here we want to mention two approaches which have proved to be useful for our application, too: the alignment method and geometric hashing.
The alignment method is based on the observation that any similarity transformation is determined up to reflection by the mapping of a single line segment. Thus, we will choose two points a, b in the pattern P and map the edge $\overline{ab}$ to all edges $\overline{uv}$ in the target set T. For each mapping we check whether it induces a (large partial approximate) matching of P. Note, in the special situation of partial matchings it is not sufficient to consider only one edge $\overline{ab}$. So in a worst case scenario one has to map all pattern edges to all edges in T. The situation is much easier to handle if scalings are not allowed (or strongly restricted). Then for a given edge $\overline{ab}$ it is sufficient to search for all target edges of approximately the same length. Analogously, if rotations are forbidden it suffices to search for all target edges with approximately the same slope as $\overline{ab}$.
An essential speed up of the alignment method can be obtained if the points in both the pattern and the target are labeled with positive values (intensities) such that for any valid matching the intensity ordering of the pattern is consistent with the intensity ordering in the target. Then the quadratic size search space of all target edges can under certain circumstances be reduced to the set of all edges in the history of the incremental Delaunay triangulation of T which has expected linear size. The details of this idea will be discussed in the next section.
The main drawback of the alignment method consists in the waste of time caused by the fact that one tries to construct

*Part of a joint research project with Deutsches Herzzentrum Berlin, supported by Deutsche Forschungsgemeinschaft, grant FL 165/4–1.

†Institut für Informatik, Freie Universität Berlin, Takustr. 9, D-14195 Berlin, email:*name@inf.fu-berlin.de*

a matching for each legal pair of edges $\overline{ab}$ in P and $\overline{uv}$ in T, but finally only few of these attempts will be successful. One can avoid this by making use of geometric hashing. This method requires some preprocessing for a hash table. After that the number of pattern points which are matched by a transformation (induced by a legal edge pair) can be counted very efficiently. Finally, one has to compute the matchings only for the best transformations.

Point pattern matching has not only been studied in computational geometry. It is such a fundamental and natural task that it comes up in various fields. Of special interest for our application is the rich pattern recognition and image processing literature on the topic, compare with [17], [7], [8]. Moreover, it is no surprise that basic ideas and principles, like the alignment paradigm, have been rediscovered several times. The same applies to the use of Delaunay triangulation graphs for the matching task, see for example [13], [11].
However, the novelty of the algorithmic solution presented below is, that, firstly, the way to construct the Delaunay triangulation graph rather than the final graph itself is used for the matching process and, secondly, that the approach works in the case of noise.

1.2 2D Gel Electrophoresis

Surveying the protein components of cells is a fundamental task in molecular biology, see [19]. For this purpose the 2–dimensional gel electrophoresis technique for protein separation was introduced by O'Farrell in 1975. With a resolution separation of several thousand spots in real samples it is almost two orders of magnitude better than competing techniques available for proteins. A 2D gel is the product of two separations performed sequentially in acrylamide gel media: isoelectric focusing as the first dimension and a separation by molecular size as the second dimension. A 2D pattern of spots each representing a protein is the result of that process. Eventually, spots are detected by staining or radiographic methods.
In Figure 1 two typical examples of 2–dimensional spot images are shown. The left image shows a gel image of a human heart ventricle tissue sample. It contains about 1500 protein spots, while the right one is an image produced in another laboratory with the same technology but with a total of about 3000 spots. Their original size is about 23cm by 29cm.
Comparing visually such an image with a master gel image (which is available in different databases on the Internet) is one way for putative protein spot identification without using expensive sequencing techniques. Another interesting possible application is related to the fact that with some diseases there are associated typical deviations of certain protein spots compared to standard spot size/intensity, an example is described in [14]. To detect such deviations is of great importance for example in view of a possible drug design.

However, for the visual comparison substantial difficulties arise from the fact that images are – due to inaccuracies in the complicated electrophoresis process itself – distorted and noisy. They have usually (especially in the inter–laboratory comparison) different separation resolution, different spot expression etc. To a much greater extent this applies to the computer assisted comparison, where one first applies an algorithm to detect spots and to extract their features like spot size, spot intensity, or spot shape, see [4] for definitions. This so called spot detection stage is a necessary preprocessing step but, at the same time, a severe error source for the subsequent spot matching problem.
Starting with the early eighties there is an extensive literature on solutions for a computer assisted gel image comparison, see [19]. However, many of the proposed algorithms make use of so–called *landmarks* and a general alignment of the images by warping techniques, [12]. Landmarks are spot pairs interactively marked in both images by the user and selected as putative matching pairs. Using various heuristics one then algorithmically extends this partial relation to a full matching.
One can do the spot matching either relying primarily on the pixel level information (like the Melanie software system [4] using spot areas) or on derived geometric information like in our solution.
For the purpose of illustration in Figure 2 more details are shown of the small rectangular window regions marked in Figure 1.

2 Our Approach to the Matching Problem; Basic Algorithmic Ideas

We address the following algorithmic *Local Matching Problem*:

Given a local spot pattern P selected from a 2D gel source image S find all local spot patterns in a target image T that resemble at least partially both the geometric shape and the spot intensities of P.

The algorithm we are going to describe starts from two assumptions. Firstly, the geometry of spot patterns is given by point patterns; and a single real value represents both spot size and spot intensity. Secondly, the algorithm should not use the relative position of the selected pattern within the source image or its location with respect to possibly given landmarks. Such information, if available, can be used to speed up our solution considerably, for example by restricting the search range in the target image.
To illustrate an instance of the local matching problem consider the example of Figure 2: Given the indicated pattern of eight spots drawn from the small rectangle in the left image find all its matching counterparts in the right image. In Figure 3 the spot point sets (without intensities) and, again, one partial matching corresponding to Figure 2 is shown. Remark that the algorithm computed a partial matching on six

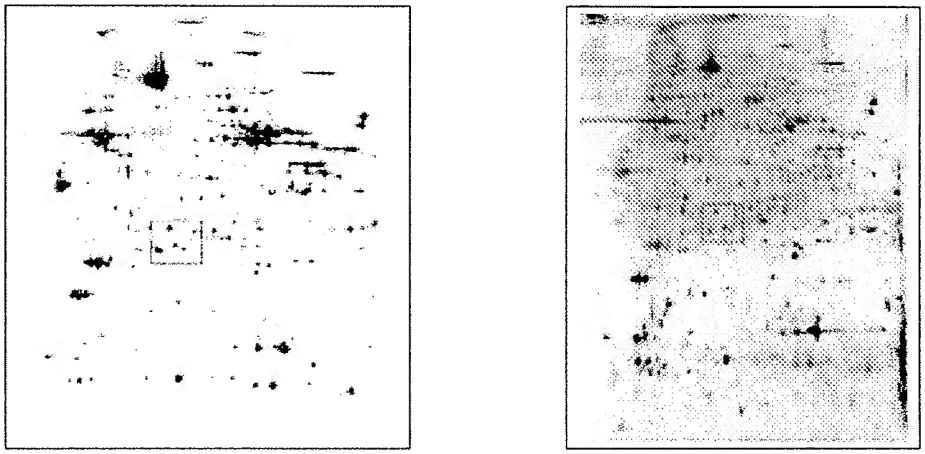

Figure 1: Two gel images of comparable samples

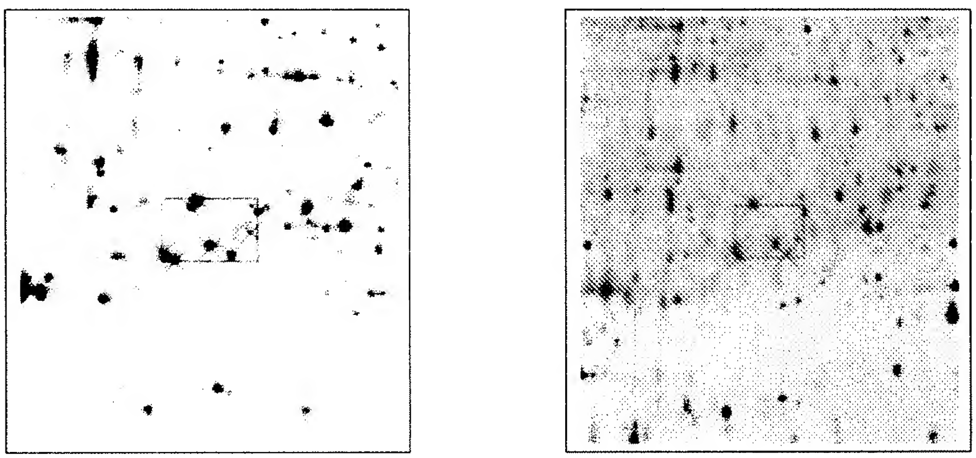

Figure 2: Detailed local images of Fig.1, a selected pattern on the left side and a partial matching

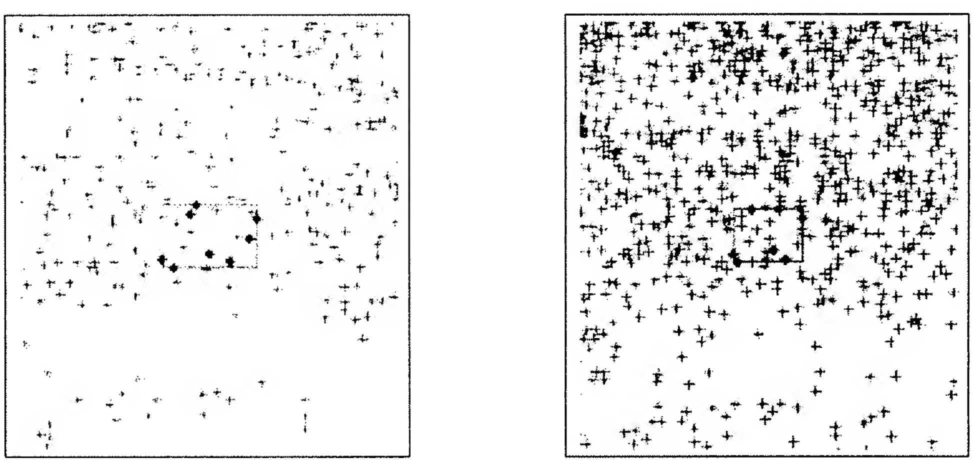

Figure 3: Spot point sets detected with indicated partial matching

spots only although there are candidate spots for the remaining two pattern spots. The reason is that their intensities differ too much from those of the pattern spots.
The problem is formalized as follows. During a first feature extracting preprocessing step a spot detection algorithm is applied to both source and target image. As a result we obtain for each gel image lists of spots. Now a spot is simply a vector $(x(s), y(s), i(s))$ consisting of its nonnegative point coordinates $(x(s), y(s))$ in the Euclidean plane and a positive number $i(s)$ describing its intensity. Observe that with this special representation we necessarily lose a lot of visual information the original gel image carries like geometric spot shapes etc. Moreover, source and target image are assumed to have the same bounding box, otherwise they are linearly scaled accordingly. The spot intensities induce a linear order in the spot list.

Next we fix the admissible transformation space and a distance measure to evaluate matchings based on the following observations.

1. Assume we want to choose a pattern P from a small rectangular window in the source image S. Source and target image can have significantly different spot numbers but since intensive spots tend to appear first it makes only sense to choose and restrict oneself to such patterns P that consist of the locally most intensive spots.

2. On the other side, a matching pattern P' in the target T should also consist of locally intensive spots. Moreover, we should also accept solutions in which P' resembles only a large portion of P. This way we can also try to correct certain errors made by the spot detection algorithm, which tends to have difficulties to interpret spots that are very close to each other correctly.

3. To model the pure geometric resemblance between P and P' we use the following simple rule. We call two line segments $\overline{st}$ and $\overline{s't'}$ (λ, α)*–similar* if their absolute slope difference is smaller than α and for their lengths we have:

$$1 - \lambda \leq |\overline{st}|/|\overline{s't'}| \leq 1 + \lambda$$

   Two point patterns P and P' are (λ, α)–similar if there is a bijection f between the point sets such that $\overline{st}$ and $\overline{f(s)f(t)}$ are (λ, α)–similar for all $s, t \in P$.
   In sum, from the application point of view we want to find (λ, α)–matchings between as large as possible subpatterns $P'' \subset P$ and target patterns P', compare also [13].

4. To model the intensity resemblance between spots we do not use directly the absolute intensity values. Instead, we apply the following very robust heuristic ranking rule that assigns to each spot a discrete intensity integer between 1 and 10. The 500 most intensive spots in an image are distributed equally according to cardinality between 10 and 6; the remaining spots are assigned to values ≤ 5 such that the total intensity sum in each class is the same. For the matching we use the criterion that a pattern spot s can only be matched to a spot in P' if their discrete intensities differ by at most 2. (This is the default value in the implementation.)

5. Since the edge similarity constants λ and α are small we know each (λ, α)–matching between $P'' \subset P$ and P' is close to a translation t, more exactly the Hausdorff distance $\tilde{H}(t(P''), P')$ between the translated P'' and P' is in worst case bounded from above by $max_{s,t \in P''} \epsilon |\overline{st}|$ with $\epsilon = \sqrt{2(1+\lambda)(1-\cos\alpha) + \lambda^2}$ in worst case.

Besides the size of P'' another criterion for evaluating the matching could be the Euclidean distance of the center of P' from the expected center position of the transformed P in the target image, provided its position in the source is known.

Given this general setting our local matching algorithm is based on the following key idea that was first time used in [18].
Let's call a triple of spots in a gel image *intensive* if its circumcircle does not contain a spot that is more intensive. An edge connecting spots s, t is *intensive* if there exists a third spot forming together with s and t an intensive triple.
This concept of intensive edges is very strongly related to the Delaunay triangulation construction of a point set. A triangulation of a point set S in the plane is called *Delaunay triangulation* if for each triangle in the triangulation its circumcircle contains only the three triangle points. One can construct such a triangulation in an incremental way by adding one point after the other, compare [9]. Now the main observation in [18] reads in our terminology:

Proposition 1: *Assume that the Delaunay triangulation of a gel image is computed incrementally by inserting spots in order of decreasing intensity. Then the set of all Delaunay triangles and edges occurring during the history of that process is exactly the set of intensive triangles and intensive edges.*

Let Hist(T) be a data structure representing all intensive edges. Its usage for the matching problem stems from the following observation.
If a pattern P of locally intensive spots occurs in T, then we can expect that, despite the possible noise, at least a few of the edges connecting spots in P will be (λ, α)–similar to edges in Hist(T).
This is the point where our approach and that one from [18] branch. While in [18] according to the alignment technique one tries to extend each occurrence of a pair of similar edges to a matching of the complete pattern we have to opt for a different strategy. The main reason for this is the small but nevertheless considerable length and slope tolerance (in the

implementation the default values are $\lambda = 0.2$ and $\alpha = 0.2$) that imply a search range that is too large.

The alternative to the alignment technique in [18] that we choose is a 2–step variant of geometric hashing , see [2].
In a first step we want to compute all locations within the target image where a good matching with the pattern is likely to occur. The actual local matchings are computed subsequently in a second step.
As pointed out above, if there is a good partial matching between pattern P and a pattern P' in T then a portion of edges connecting spots in P will have (λ, α)–similar matching edges in Hist(T).
Even more, if we associate with each occurrence of a similar edge the translation vector of the edge midpoints then a matching pattern will correspond to a cluster of vectors in the translation space. It turns out that the clusters can be computed and evaluated efficiently. For the best candidate clusters we then recompute locally an actual matching.
Another problem to be addressed is how to proceed in the case that there is a more severe distortion between the pattern P and its counterpart P' in the target image.

3 Details of the Local Matching Algorithm

3.1 Preprocessing the Target Image

As indicated above we triangulate the underlying point set of T using the incremental Delaunay triangulation algorithm of [9] inserting spots according to decreasing absolute spot intensities. When a new spot is inserted a few edges are deleted from the current triangulation, a few new ones added. We call these edges Delaunay edges. Additionally we consider all edges connecting the new spot with opposite spots in neighboring triangles. Let us call these edges *flipped diagonals*. We store all Delaunay edges and flipped diagonals occurring during that process in a data structure Hist$^*(T)$, that describes the *extended history* of the incremental Delaunay triangulation of T. With each object in Hist$^*(T)$ we store also its length and its slope. For this purpose range trees are the appropriate -at least theoretically- data structure, see for example [5].

Using Seidel's backward analysis technique, see [16], we have:

Proposition 2: *The expected number of edges in the extended history of a randomized incremental Delaunay triangulation of a point set in the plane is bounded by $12n$, where n is the number of points.*

The main question is to which extent do the history, respectively extended history allow to recognize a local pattern ? Assume we are given an identical copy P' of the pattern P in the target. The matching between P and P' is done via Delaunay edges connecting spots of P'. However, how many edges actually belong to the history depends not only on P' but also on its context in the target, since the 'local' history of the Delaunay triangulation can be strongly influenced by intensive spots in the neighborhood of P', as demonstrated by the example in Figure 4. In Figure 4 let c, d and f be spots in the target triangle pattern P' corresponding to pattern spots in P. The pattern was chosen from a window corresponding to the box drawn with dashed lines. Now consider the incremental Delaunay triangulation which inserts the spots in alphabetical order. We observe that none of the edges forming the pattern triangle occurs in the history of the triangulation, but all of them are flipped diagonals in Hist$^*(T)$. Therefore, using Hist$^*(T)$ as search space we still have the advantage of its expected linear size and at the same time we increase the probability to include edges from P'.

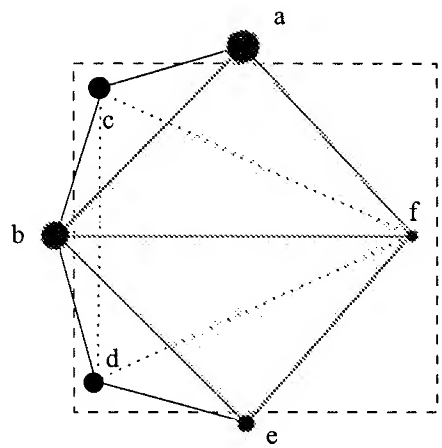

Figure 4: Pattern edges that are not Delaunay edges

This also applies to the case that there is some noise in the target. To illustrate these facts we have run computer experiments that mimic the situation and quantities given in our approach to the local gel image matching. Assume we randomly draw from a unit square B a pattern P' of k ($k = 8, k = 12$) spots. Moreover, we take a 7×7 square T that contains B and generate $48k$ random points in $T \setminus B$. Next we simulate overall noise in T by adding l new random points. Finally, by picking at random a permutation of all $49k + l$ points in T we fix a linear intensity order. For each value of l we compute 1000 random instances and count in each corresponding incremental Delaunay triangulation the fraction of those edges in the simple, respectively extended history that connect pattern spots. In Figure 5 we summarize the data which clearly indicate that Hist$^*(T)$ is well suited for identifying a local pattern in images with up to one third of noisy points.

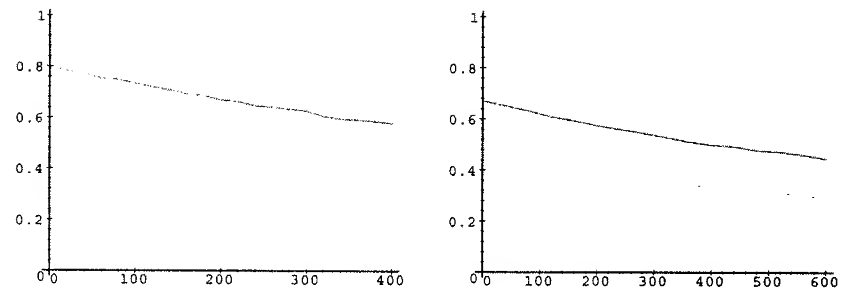

Figure 5: The fraction of pattern edges that are elements of Hist(T) (lower curves) and Hist$^*(T)$ in the presence of l random distortion spots (abscissa) and random intensity order for $k = 8$ (left) and $k = 12$

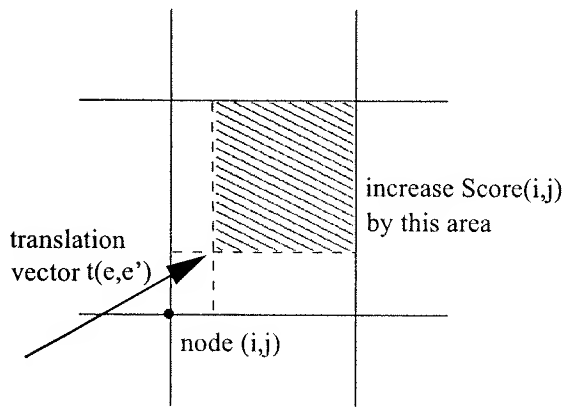

Figure 6: Updating the score for a pair e, e' of similar edges

3.2 Approximating the Matching Locations

After the preprocessing we are able to answer queries of the type: Given a pattern edge e find all target edges e' in Hist$^*(T)$ that meet the tolerance bounds with respect to length, slope, and discrete spot intensities.
However, we do not store the results of such a query as an edge list. We first compute the vector $t(e, e')$ that translates the midpoint of edge e to the midpoint of e'. For all these we maintain a scoring list that indirectly stores translation vectors and yields at the same time clusters of such vectors. Observe that such clusters correspond to possible matching locations. This is done as follows.
The bounding box of T is interpreted as the possible space for translation vectors $t(e, e')$. Next we overlay a regularly spaced grid on the translation space and maintain a data structure for integer scores, initially all zero, which are defined for each grid node.
Each translation vector $t(e, e')$ increases the score of the four grid nodes defining the grid cell the vector falls into. $t(e, e')$ subdivides this cell into four rectangles as depicted in Figure 6. Each of the four grid nodes adds to its current score an amount proportional to the area of the opposite rectangle given the total area by 100. Let Score(i, j) be the total score accumulated in grid node (i, j) after probing all $\binom{|P|}{2}$ pattern edges. All local maxima that are greater than a threshold value depending on $|P|$ are considered to correspond to potential matching locations.
Eventually, we can approximate the actual center (i_c, j_c) of the vector cluster stemming from a local maximum at node (i, j) by computing a weighted average of the scores at (i, j) and all scores at neighboring grid nodes:

$$(i_c, j_c) = \frac{\sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \text{Score}(k, l) \cdot (k, l)}{\sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \text{Score}(k, l)}$$

Figure 7 illustrates the result of the scoring procedure in the neighborhood of the actual matching position in the target in Figure 2.

3.3 Verifying and Evaluating a Local Matching

After the scoring procedure we are given a list of putative locations of matching pattern centers c. Next we recompute the actual patterns that define the matchings. To this end we consider the bounding box of the pattern P, scale it by length tolerance factor $1 + \lambda$ and for each center c we compute the rectangular sub-image T_c centered at c of the target image with this size.
Next we have a *voting* procedure to compute a partial (λ, α)–matching between P and some pattern P' in T_c. This voting is very similar to the scoring procedure above. We compute the extended history Hist$^*(T_c)$. For each pattern edges $\overline{st}$ in P we search for all (λ, α)–similar edges $\overline{s't'} \in$ Hist$^*(T_c)$. But this time we insert each found spot s' in a *candidate list* for s and t' in a corresponding list for t. We say that s gets a vote from s' and t from t', respectively.

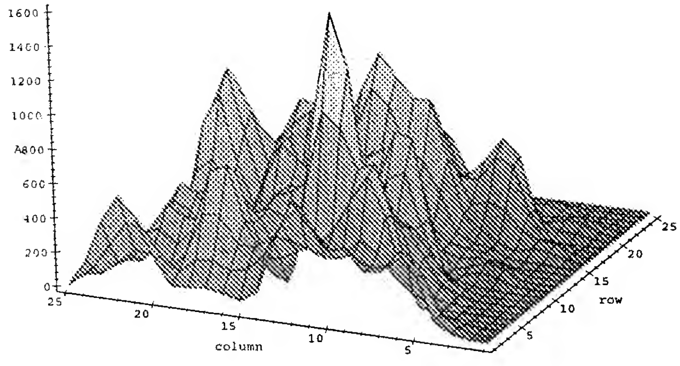

Figure 7: Local result of the scoring procedure

Eventually, we form a tentative partial matching between all pattern spots s and their matching candidates that accumulated a maximum number of votes and exceed a certain threshold depending on the pattern size.
This tentative matching is neither necessarily a 1-1–matching nor a (λ, α)–matching. Such a situation especially occurs if two pattern spots s, t are very close to each other but in the target there is simply only one sufficiently intensive spot in that corresponding place, or there is a spot pair s', t' that violates the slope tolerance. Therefore there is a final *clearing* step that chooses a maximum size sub-matching of the tentative matching that is both 1–1 and meets the tolerance bounds.

How should we evaluate a found local matching? This is clearly application depending. If source and target gel image represent non–comparable samples then the matching cardinality is the only criterion. Otherwise, a ranking that combines both the cardinality and the distance from the expected location is possible. In the latter case we have also the possibility to test the consistency of a local matching result by the following simple iterative method. We accept a local matching for P only if it is confirmed by local matchings for neighboring patterns Q that have nontrivial intersections with P. This idea serves as the basis for a *global matching* algorithm, see below.

3.4 Strongly Distorted Patterns

The local matching algorithm described before is robust to the respect that it indeed computes for each pattern a list of partial matchings within the prescribed tolerance bounds. Another problem is how to proceed in the case that there is a more severe geometric distortion between the pattern P and its counterpart P' in the target image. Obviously, one solution would be to increase the values for slope and length tolerances yielding increased time bounds for the geometric hashing process. What we are doing instead is a standard heuristic trick, compare [17]; we distort the pattern P iteratively in a typical (application depending) way and then we search for the distorted pattern while keeping similarity tolerances small. In our case these distortions are combinations of independent $x - y$–scalings and shifts that transform rectangular regions into parallelograms. Let us denote by Distort(P) the list of all patterns derived from P that way.
The other more elegant way is the following. We consider the 4–dimensional transformation space of translations and x– resp. y–scalings. Given upper bounds on the single coordinates we can implement a scoring procedure that accumulates scores in nodes of a sufficiently refined 4–dimensional grid for each pair of similar edges.

3.5 Estimating the Time Complexity

First of all, we have to remark that the results of the following worst case analysis do not completely reflect the performance of the algorithms for real world input data. However, this analysis is useful in order to compare our algorithm with previous approaches and to make clear where the progress comes from.
Let k and n denote the sizes of the pattern P and of the target point set T.
The running time of the alignment method is the product of the number $sep(P,T)$ of (λ, α)–similar edge pairs and the costs $m_t(P,T)$ to compute a matching under a translation t induced by such an edge pair. In the general setting we have a trivial but tight worst case upper bound of $sep(P,T) = \mathcal{O}(k^2 n^2)$, compare also page 153ff. in [1], and $m_t(P,T) = \mathcal{O}(k \log n)$. Switching to the extended Delaunay history Hist* we get an expected value of $sep(P,T) = \mathcal{O}(k^2 n)$.

This implies a total expected upper bound of $\mathcal{O}(k^3 n \log n)$ which was obtained in [18]. Our geometric hashing variant (i. e., the voting procedure) allows to remove the $k \log n$ factor and to replace it by some additive terms. The general upper bound for geometric hashing of $\mathcal{O}(n^3)$ is also very bad, even without taking into account the preprocessing and final computation of the matching.
In contrast, our voting procedure requires $\mathcal{O}(n \log n)+$ $+\mathcal{O}(k^2 n) + \mathcal{O}(|G|)$ time, where the first term represents the costs of the incremental Delaunay triangulation, while the second one bounds the number of similar edge pairs. In the third term $|G|$ denotes the constant size of the grid used to store the votes. It is of smaller order and can be ignored. Finally, computing the C best matchings costs $C \cdot m_t(P,T)$. Thus, altogether for computing the best matchings for one given pattern we achieve an expected $\mathcal{O}(n(\log n + k^2))$ upper bound which has to be multiplied by the number of patterns in Distort(P).

3.6 Global Matching via Local Matching

Previous algorithms for the global matchings of gel images are based mainly on landmarks set by the user. In this context a landmark is a pair of points, one in the source image S, the other one in the target T. Thus, the user fixes a partial matching on a sufficiently large set of so called support points $S_{supp} \subset S$. Triangulating S_{supp} and constructing the corresponding triangulation in the target image one gets a piecewise affine transformation f defined on the triangles. Finally, for any source point $p \in S$ one has to search for the nearest neighbor of $f(p)$ in T where the distance is a combination of the euclidian distance and the intensity difference.
Our aim is to avoid interactive landmark setting by making use of the local matching solution. The problem is that this algorithm computes several matchings of a chosen pattern and in general it is not clear which is the right one. There are two approaches to improve the confidence in a found matching. The first idea is to insert a spot $p \in S$ into different patterns $P_1, \ldots, P_m$ and to compute their best local matchings independently. We considered four patterns extending a bounding box from p into the four quadrants. Then a point $q \in T$ will be accepted as an image of p if for all patterns there is at least one (of the best) matching mapping p to q. It turns out that answers found this way are sufficiently satisfying. However, due to the strong restrictions within this approach it may happen that no answer will be replied.
The second approach consists in covering the source image by patterns in a grid like fashion. Computing the best local matchings for all patterns one has to look for a consistent choice of matchings. Let P_1 and P_2 be neighboring patterns in S. A matching of P_1 to P_1' in T will be called consistent with a matching of P_2 to P_2' if for their centers $c(P)$ holds that the edges $\overline{c(P_1), c(P_2)}$ and $\overline{c(P_1'), c(P_2')}$ are (λ', α')-similar. Here, we have to enlarge the tolerance bounds to reflect the fact that for each single pattern P_i the matching can stem from the list Distort(P_i).

3.7 Implementation and User Interface

The matching algorithms have been implemented and are part of the Carol software system [3]. It has essentially two parts:
The first part, the combinatorial and geometrical kernel of the matching algorithms, has been implemented in C++. It makes essential use of the Standard Template Library (STL) and of the Computational Geometry Algorithms Library (CGAL), [6]. The latter library provides several geometric data structures and functions and especially an implementation of the incremental Delaunay triangulation. The second part of the Carol system is the graphical user interface which has been implemented in Java. It can be run as an applet started out of an internet browser or as an application. The communication with the algorithmical program part is established via internet sockets, whereby the C++-program works as a server which waits for matching requests from the Java-client, performs the computation and sends eventually back the results to the client. The program will be eligible to match gel images from databases all over the internet. This feature is strongly supported by the possibility to run the user interface as an applet and furthermore by the client-server architecture of the program.

The user has the possibility to set parameters like tolerance bounds, pattern size etc., for more details of the user interface see *http://gelmatching.inf.fu–berlin.de*.
An unavoidable and critical preprocessing step is the spot detection stage. It is planned to include into the Carol system a spot detection algorithm that has been recently developed at Deutsches Herzzentrum Berlin, see [15].

The local matching algorithm run on a Sun Sparc Ultra 1 computes the best 9 matchings for a pattern of 8 spots in about 3 seconds including the preprocessing of a 3000 spot target image. Each further pattern in the list Distort(P) increases this time on the average by about 0.3 seconds.

4 Conclusions and Directions for Further Work

We have presented the underlying ideas for an algorithmic solution of the local matching problem of 2D patterns of protein spots in electrophoresis images. Its main features are:

1. Local matchings for a source pattern are found in the target image without knowledge of its context.
2. The local matching algorithm works for locally intensive patterns. There are standard techniques like point location combined with affine approximation and nearest neighbor search that extend the solution to other spots.
3. The local matching algorithm can be used as a basic step for the global matching problem for gel images. In fact, local matching is used then like landmark setting.

4. The central idea for the algorithm stems from the use of the extended history of the incremental Delaunay triangulation, which proved to be a suitable structure for the local matching problem because of its expected linear size and its robustness in the presence of noise.

There are several issues for theoretical investigations raised by our approach. One topic for further work is certainly the analysis of the 'local' history of a random incremental Delaunay triangulation and its dependency on noise.
Last, but not least, there is the question for other applications of our local matching algorithm which uses only a minimum of specific application knowledge.

Acknowledgement: We would like to thank Klaus–Peter Pleißner from Deutsches Herzzentrum Berlin, Helmut Alt, Christian Knauer and Sven Schönherr from Freie Universität Berlin for their generous help.

References

[1] P. Agarwal, J. Pach, Combinatorial Geometry, John Wiley 1995

[2] H. Alt, L. Guibas, Discrete Geometric Shapes: Matching, Interpolation, and Approximation, A Survey, TR B 96-11, Inst. f. Informatik, Freie Universität Berlin, to appear in J. Urrutia, J.–R. Sack, eds., Handbook for Computational Geometry (North Holland)

[3] H. Alt, F. Hoffmann, K. Kriegel, C. Wenk, K.-P. Pleißner, CAROL - New Algorithmic Tools for Comparing Two–Dimensional Electrophoretic Gel Images, Proceedings Electrophorese Forum '97, Strasbourg, P21

[4] R. Appel, J. Vargas, P. Palagi, D. Walther, D. Hochstrasser, Melanie II - a third generation software package for analysis of two–dimensional electrophoresis images: II. Algorithms, Electrophoresis 18 (1997), 2735–2748

[5] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, Springer Verlag, 1997

[6] CGAL, The Computational Geometry Algorithms Library, http://www.cs.ruu.nl/CGAL

[7] S.–H. Chang, F.–H. Cheng, W.–H. Hsu, G.–Z. Wu, Fast Algorithm for Point Pattern Matching: Invariant to Translation, Rotations and Scale Changes, Pattern Recognition 30(2) (1997), 311–320

[8] F.–H. Cheng, Point Pattern Matching Algorithm Invariant to Geometrical Transformation and Distortion, Pattern Recognition Letters 17 (1997), 1429–1435

[9] L. Guibas, D. Knuth and M. Sharir, Randomized Incremental Construction of Delaunay and Voronoi Diagrams, Algorithmica 7(4) (1992) 381–413

[10] D. P. Huttenlocher, G. Klanderman and W Rucklidge, Comparing images using the Hausdorff distance, IEEE Trans. Pattern Analysis and Machine Intelligence 15 (1993) 850–863

[11] A. Finch, R. Wilson, E. Hancock, Matching Delaunay Graphs, Pattern Recognition 30(1) (1997), 123–140

[12] P. F. Lembkin, Comparing two–dimensional electrophoretic gel images across the Internet, Electrophoresis 18(3–4) (1997), 461–470

[13] H. Ogawa, Labeled Point Pattern Matching by Delaunay Triangulation and Maximal Cliques, Pattern Recognition 19(1) (1986), 35–40

[14] K.–P. Pleissner, V. Regitz–Zagrosek, C. Weise, M. Neuss, B. Kruedewagen, P. Soeding, K. Buchner, F. Hucho, A. Hildebrandt, E. Fleck, Chamber–specific expression of human myocardial proteins detected by two–dimensional gel electrophoresis, Electrophoresis 16 (1995) 841–850

[15] K.–P. Pleissner, A. Sahlström, S. Wegner, H. Oswald, E. Fleck, Protein spot detection in WWW–accessible 2–D gel images using the Watershed Transformation, Proceedings Electrophorese Forum '97, Strasbourg, P20

[16] R. Seidel, Backwards Analysis of Randomized Geometric Constructions, in J. Pach, ed., New Trends in Discrete and Computational Geometry (Springer Verlag), 1993

[17] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis and Machine Vision, Chapman and Hall, 1994

[18] G. Weber, L. Knipping and H. Alt, Point Pattern Matching in Astronautics, J.Symbolic Computation 17 (1994) 321–340

[19] M. R. Wilkins, K. L. Williams, R. D. Appel, D. F. Hochstrasser (eds.) Proteome Research: New Frontiers in Functional Genomics, Springer-Verlag Berlin Heidelberg New York, 1997